**Санкт-Петербургский государственный университет**

**Р А Б О Ч А Я П Р О Г Р А М М А**

**УЧЕБНОЙ ДИСЦИПЛИНЫ**

Функциональное программирование (Lisp, Haskell, F#)

Functional Programming (Lisp, Haskell, F#)

**Язык(и) обучения**

русский

Трудоемкость в зачетных единицах: 2

Регистрационный номер рабочей программы: 051774

**Раздел 1. Характеристики учебных занятий**

**1.1. Цели и задачи учебных занятий**

Дисциплина «Функциональное программирование» входит в перечень дисциплин, формирующих подготовку специалиста в области математико-информационных наук. Она представляет собой комплекс знаний, умений и навыков, применимых как и для овладения математическими методами в области информационных технологий, так и для развития у учащихся навыков решения практических задач.

Целью дисциплины является обучение основным принципам функционального программирования и математическим понятиям, лежащим в основе данной парадигмы программирования. Целью дисциплины является также получение обуучающимися практических навыков программирования и навыков применения принципов функционального программирования при решении реальных задач.

Основной задачей практических занятий, предусматриваемых учебным планом, является изучение основных принципов функционального программирования, приемов, используемых при программировании на функциональных языках, примеров практического применения этих приемов, основных понятий лямбда исчисления и основных математических результатов, полученных в данной области. Ставится также задача освоения студентами языка функционального программирования Хаскелл и развитие навыков самостоятельного решения задач на этом языке.

**1.2. Требования подготовленности обучающегося к освоению содержания учебных занятий (пререквизиты)**

Программа дисциплины рассчитана на обучающихся 3–ого курса. Максимальная эффективность Программы будет обеспечена, если обучающийся владеет базовыми математическими понятиями, достаточными для понимания математических утверждений. Необходимо владение навыками программирования с использованием процедурной и объектно-ориентированной парадигм, владение основными понятиями теории алгоритмов.

**1.3. Перечень результатов обучения (learning outcomes)**

По окончании обучения обучающийся должен:

• знать содержание дисциплины «Функциональное программирование» и иметь достаточно полное представление о возможностях применения её разделов в различных прикладных областях науки и техники;

• уметь применять методы функционального программирования для решения разнообразных задач в области информатики.

**1.4. Перечень и объём активных и интерактивных форм учебных занятий**

Индивидуальная работа с доступными текстами в данной предметной области, а также удовлетворение личных познавательных потребностей.

**Раздел 2. Организация, структура и содержание учебных занятий**

**2.1. Организация учебных занятий**

**2.1.1 Основной курс**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Трудоёмкость, объёмы учебной работы и наполняемость групп обучающихся | | | | | | | | | | | | | | | | | | |
| Код модуля в составе дисциплины,  практики и т.п. | Контактная работа обучающихся с преподавателем | | | | | | | | | | | | Самостоятельная работа | | | | Объём активных и интерактивных  форм учебных занятий | Трудоёмкость |
| лекции | семинары | консультации | практические  занятия | лабораторные работы | контрольные работы | коллоквиумы | текущий контроль | промежуточная  аттестация | итоговая аттестация | под руководством преподавателя | в присутствии  преподавателя | сам. раб. с использованием  методических материалов | текущий контроль (сам.раб.) | промежуточная аттестация (сам.раб.) | итоговая аттестация  (сам.раб.) |
| ОСНОВНАЯ ТРАЕКТОРИЯ | | | | | | | | | | | | | | | | | | |
| Форма обучения: очная | | | | | | | | | | | | | | | | | | |
| Семестр 5 |  |  |  | 30 |  |  |  |  | 2 |  |  | 30 |  |  | 10 |  | 10 | 2 |
|  |  |  |  | 10-25 |  |  |  |  | 10-25 |  |  | 10-25 |  |  | 1-1 |  |  |  |
| ИТОГО |  |  |  | 30 |  |  |  |  | 2 |  |  | 30 |  |  | 10 |  |  | 2 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Виды, формы и сроки текущего контроля успеваемости и промежуточной аттестации | | | | | | |
| Код модуля в составе дисциплины, практики и т.п. | Формы текущего контроля успеваемости | | Виды промежуточной аттестации | | Виды итоговой аттестации  (только для программ итоговой аттестации и дополнительных образовательных программ) | |
| Формы | Сроки | Виды | Сроки | Виды | Сроки |
| ОСНОВНАЯ ТРАЕКТОРИЯ | | | | | | |
| Форма обучения: очная | | | | | | |
| Семестр 5 |  |  | зачёт, устно, традиционная форма | по графику промежуточной аттестации |  |  |

**2.2. Структура и содержание учебных занятий**

|  |  |  |  |
| --- | --- | --- | --- |
| **№ п/п** | **Наименование темы (раздела, части)** | **Вид учебных занятий** учебных занятий | **Кол-во часов** |
| 1 | Понятие функционального программирования. История создания и развития функционального программирования. Сравнение функционального программирования и других парадигм программирования. Понятие чисто функционального языка. Понятие прозрачности по ссылкам (referential transparency) | практические занятия | 2 |
| сам. раб. в присутствии преподавателя | 2 |
| 2 | Основные возможности языка Хаскелл. Создание простых программ на языке Хаскелл. Понятие хвостовой рекурсии. Накапливающие параметры. Более сложные возможности языка Хаскелл: конструкции let, where, анонимные переменные, синтаксис, управляемый отступами. | практические занятия | 2 |
| сам. раб. в присутствии преподавателя | 2 |
| 3 | Списки. Основные функции для работы со списками. Понятие структурной индукции. Понятие функции высшего порядка. Примеры таких функций. Лямбда-выражения и примеры их использования. Основные стандартные функции высшего порядка. Функция свертки. Кортежи. Алгебраические типы данных. Пример: работа с деревьями. | практические занятия | 2 |
| сам. раб. в присутствии преподавателя | 2 |
| 4 | Функции, возвращающие функции в качестве результата. Композиция функций. Понятие комбинатора. Частичная параметризация. Карринг. Понятие статическое связывания. Замыкания. Замыкания в не функциональных языках (C#, C++). | практические занятия | 4 |
| сам. раб. в присутствии преподавателя | 4 |
| 5 | Ленивые вычисления. Передача параметров «по требованию». Бесконечные списки и бесконечные структуры. Рекурсивные соотношения между бесконечными списками. Аналоги ленивых вычислений и бесконечных списков в других программных парадигмах. | практические занятия | 4 |
| сам. раб. в присутствии преподавателяа | 4 |
| 6 | Система типов языка Хаскелл. Автоматический вывод типов. Алгоритм Хиндли-Милнера. Понятие класса. Полиморфизм в языке Хаскелл и сравнение его с возможностями полиморфизма в других языках программирования. | практические занятия | 4 |
| сам. раб. в присутствии преподавателя | 4 |
| 7 | Обработка ошибок в языке Хаскелл. Тип Maybe. Классы Functor и Traversable. Функция >>=. Конструкции do. Понятие монады. Проблема реализация ввод-вывода в строго функциональных языках. Ввод-вывод в языке Хаскелл. | практические занятия | 46 |
| сам. раб. в присутствии преподавателя | 42 |
| 8 | Лямбда-исчисление. Представление чисел с помощью лямбда выражений. Бета-редукция. Понятие нормальной формы. Понятие конфлюентности и теорема о единственности нормальной формы. Нормальный и аппликативный порядок редукций. | практические занятия | 4 |
| сам. раб. в присутствии преподавателя | 4 |
| 9 | Изоморфизм Карри-Ховарда. Теорема Рейнольдса о параметризации и примеры ее применения. | практические занятия | 4 |
| сам. раб. в присутствии преподавателя | 4 |
| 10 | Промежуточная аттестация | самостоятельная работа | 10 |
| зачет | 2 |
| **Итого** | | | **72** |

**Раздел 3. Обеспечение учебных занятий**

**3.1. Методическое обеспечение**

**3.1.1 Методические указания по освоению дисциплины**

Освоение дисциплины состоит в разборе основных понятий, принципов и типичных задач по темам курса и решения практических заданий с использованием компьютера. Методические материалы включают в себя следующие типы материалов — учебники, учебные пособия, методические указания для учащихся. Интернет-ресурсы, электронные учебные пособия, с опорой на которые проводится аудиторная работа.

**3.1.2 Методическое обеспечение самостоятельной работы**

Основная и дополнительная литература

**3.1.3 Методика проведения текущего контроля успеваемости и промежуточной аттестации и критерии оценивания**

Промежуточная аттестация проводится в виде письменной работы. Работа состоит из 10 заданий. Для получения «Зачёт» обучающейся должен выполнить 8 из них. От 5 до 7 правильных работ, даются дополнительные письменные работы от 5 до 3.

«Незачет» ставится, если обучающийся выполнил меньше 5 письменных заданий.

**3.1.4 Методические материалы для проведения текущего контроля успеваемости и промежуточной аттестации (контрольно-измерительные материалы, оценочные средства)**

Примерные вопросы для промежуточной аттестации:

1. Что такое хвостовая рекурсия? Что такое накапливающие параметры? Пример их использования.

2. Что означает понятие referential transparency (прозрачность по ссылкам)?

3. Что имеется в виду, когда говорят, что в Haskell используется двумерный синтаксис (off-side rule)? Опишите его правила.

4. Как в Haskell'e определить тип, параметризованный другим типом? Определите тип Tree (дерево), который для любого типа определяет дерево, содержащее значения этого типа.

5. Что такое лямбда выражение? Приведите пример его использования.

6. Что такое карриннг (currying)? Что имеется в виду, когда говориться что все функции в Haskell имеют один параметр?

7. Что такое сечение (section) в Haskell'е? Приведите пример использования section.

8. Нелокальные переменные в определении функции. В какой момент они получают значения? Приведите какой-нибудь пример, когда это имеет значение. Что такое статическое связывание?

9. Что в функциональном программировании называют замыканием? Приведите пример.

10. Что делают функции foldr и foldl ? Чем они отличаются? Приведите определение (код) одной из этих функций и пример ее использования.

12. Как можно определить функцию, аналогичную foldr, для деревьев? Что такое катаморфизм?

13. Перечислите конструкции, которые можно использовать в list comprehension. Приведите примеры их использования.

14. Что означает, что в Haskell'е реализован 'ленивый порядок вычислений' (lazy evaluation). Покажите на каком-нибудь примере, чем порядок вычислений при lazy evaluation отличается от обычного.

15. Что означает, что в Haskell умеет работать с бесконечными структурами данных? Приведите какой-нибудь пример, того, как на Haskell можно описать 'бесконечную структуру'.

16. Какой прием программирования на Haskell'е называется 'завязывание узлов' (tying the knots)? Приведите пример.

17. Какие конструкции и понятия из обычного (не функционального) программирования можно, в каком-то смысле, назвать аналогами ленивого вычисления в Haskell'е? Надо привести какие-нибудь два примера. Что такое идиома Copy On Write?

18. Какие типы имеют функции length, zip, map, foldr ?

19. Приведите какой-нибудь пример описания класса в Haskell и какой-нибудь пример того, как объявить instance класса.

20. Что в Haskelle означает слово deriving? Приведите пример его использования.

21. Опишите как работает автоматический вывод типа функции (алгоритм Хиндли-Милнера) на примере какой-нибудь функции.

22. Что делает функция >>= для списков? Приведите пример ее использования.

23. Как написать функцию find (поиска по условию), так, чтобы она корректно сообщала о том, что ничего не найдено? Как в Haskell'e определен тип Maybe? Приведите решение с использованием Maybe и еще какой-нибудь вариант решения.

24. Приведите пример использования конструкции do для записи последовательности вычислений, любое из которых может завершиться неудачно, и которые надо выполнять до первой неудачи.

25. Что такое failure continuation? Приведите пример его использования

26. Что такое continuation-passing style? Приведите пример его использования.

27. Опишите оператор <=< для функций, возвращающих пару (значение, цена). Приведите пример использования оператора.

28. Как описать ‘почти монаду’ для функций, возвращающих пару (ответ, цена). (Имеются в виду оператор >>>= и функция return1, рассмотренные на занятии).

29. Что такое класс Functor? Приведите пример того как какой-то класс объявить экземпляром Functor.

30. Что такое монады в Haskell? (Можно не формально, как вы это понимаете). Приведите какие-нибудь два примера монад.

31. Как описать ‘почти монаду’ для функций, читающих список и возвращающих пару (ответ, новый список). (имеются в виду функции >>>= и return1 с последних занятий). Приведите пример их использования.

32. Определите тип Expr с несколькими переменными и с поддержкой конструкции let и определите для него функцию, вычисляющую значение выражения.

33. Определите тип Expr с несколькими переменными и с поддержкой конструкции let и определите для него функцию, вычисляющую значение выражения. При этом надо обеспечить возможность, чтобы значения, присваемые в let выражениях вычислялись бы не тогда, когда мы задаем значение переменной, а тогда, когда мы используем переменную.

34. Как в чистом лямбда исчислении можно моделировать целые числа (числа Черча)? Приведите пример определения какой-нибудь арифметической операции (задача с занятий. про получение следующего числа или ваш собственный пример). Как по числу Черча получить соответствующее ему целое число?

35. Что такое бета-редукция? Какие сложные случаи надо учитывать при строгом определении бета-редукции?

36. Что такое нормальная форма? Верно ли, что у любого лямбда выражения существует нормальная форма? Что такое нормальный и аппликативный порядки применения редукций? Каким замечательным свойством обладает нормальный порядок?

37. Что такое конфлюентность? Докажите, что из кофлюентности следует единственность нормальной формы.

38. Что такое комбинатор неподвижной точки? Приведите пример его использования. Что такое Y комбинатор?

39. Что такое изоморфизм Карри-Ховарда?

40. Что такое «бесплатные теоремы» (free theorem)? Приведите пример «бесплатной теоремы» для какой-нибудь функции.

**3.1.5 Методические материалы для оценки обучающимися содержания и качества учебного процесса**

Оценка обучающимися содержания и качества учебного процесса по дисциплине осуществляется в установленном в СПбГУ порядке.

**3.2. Кадровое обеспечение**

**3.2.1 Образование и (или) квалификация штатных преподавателей и иных лиц, допущенных к проведению учебных занятий**

К преподаванию дисциплины могут быть допущены преподаватели, имеющие диплом о высшем образовании по соответствующему направлению.

**3.2.2 Обеспечение учебно-вспомогательным и (или) иным персоналом**

Специальных требований нет.

**3.3. Материально-техническое обеспечение**

**3.3.1 Характеристики аудиторий (помещений, мест) для проведения занятий**

Стандартная аудитория или компьютерный класс.

**3.3.2 Характеристики аудиторного оборудования, в том числе неспециализированного компьютерного оборудования и программного обеспечения общего пользования**

Стандартное оборудование.

**3.3.3 Характеристики специализированного оборудования**

Специальных требований нет.

**3.3.4 Характеристики специализированного программного обеспечения**

Специальных требований нет.

**3.3.5 Перечень и объёмы требуемых расходных материалов**

Требуются фломастеры, не менее двух фломастеров разных цветов, тряпки для вытирания досок.

**3.4. Информационное обеспечение**

**3.4.1 Список обязательной литературы**

1. Филд А., Харрисон П. Функциональное программирование. М.: Мир, 1993. 637 с.

2. Липовача М. Изучай Haskell во имя добра! ДМК-Пресс, 2012. 490 с.

**3.4.2 Список дополнительной литературы**

1. Макеев Г. Основы функционального программирования на языке Haskell. Электронный ресурс: <http://greg.southural.ru/science/MakeevGA-Haskell.pdf>.

2. Шевченко Д. О Haskell по-человечески. Электронный ресурс: <https://www.ohaskell.guide>.

3. Haskell Language. Электронный ресурс: https://www.haskell.org/documentation.

**3.4.3 Перечень иных информационных источников**

Специальных требований нет.

**Раздел 4. Разработчики программы**

Симуни Михаил Лазаревич, старший преподаватель мат-мех ф-та, каф. информатики. m.simuni@spbu.ru, +7 (921) 304 43 94, +7 (812) 428 42 33